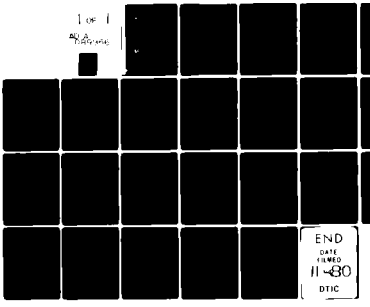
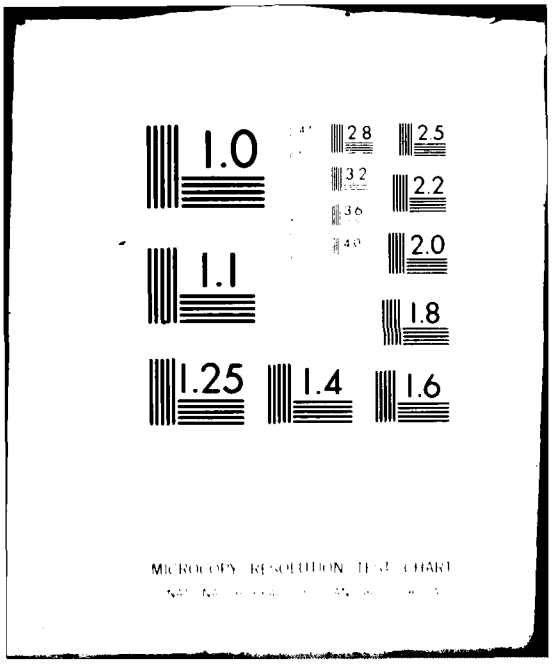
AD-A089 966 SCIENCE APPLICATIONS INC HUNTSVILLE AL F/G 9/2
INTERACTIVE TEXT EDITING FOR THE SIGMA 5.(U)
SEP 80 S K COUNSELMAN DAAK40-79-D-0023
UNCLASSIFIED SAI-81-739-HU DRSMI/RG-CR-80-8 NL

1 of 1




1.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6
MICROCOPY RESOLUTION TEST CHART

TECHNICAL REPORT RG-CR-80-8

INTERACTIVE TEXT EDITING FOR THE SIGMA 5

FINAL REPORT

Kevin Counselman
Science Applications, Inc.
2109 W. Clinton Av.
Huntsville, Al. 35805

For
Guidance and Control Directorate
U. S. Army Missile Laboratory

SEPTEMBER 1980

U.S. ARMY MISSILE COMMAND
Redstone Arsenal, Alabama 35898

APPROVED FOR PUBLIC RELEASE, DISTRIBUTION UNLIMITED

DISPOSITION INSTRUCTIONS

DESTROY THIS REPORT WHEN IT IS NO LONGER NEEDED. DO NOT RETURN IT TO THE ORIGINATOR.

DISCLAIMER

THE FINDINGS IN THIS REPORT ARE NOT TO BE CONSTRUED AS AN OFFICIAL DEPARTMENT OF THE ARMY POSITION UNLESS SO DESIGNATED BY OTHER AUTHORIZED DOCUMENTS.

TRADE NAMES

USE OF TRADE NAMES OR MANUFACTURERS IN THIS REPORT DOES NOT CONSTITUTE AN OFFICIAL ENDORSEMENT OR APPROVAL OF THE USE OF SUCH COMMERCIAL HARDWARE OR SOFTWARE.

DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

(18) DRSMI/RC

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(17) REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: RG-CR-80-8	2. GOVT ACCESSION NO.: AD-A089 966	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): Interactive Text Editing for the Sigma 5	(9) 5. TYPE OF REPORT & PERIOD COVERED: Final Technical Report
	(14) 6. PERFORMING ORG. REPORT NUMBER: SAI-81-739-HU
7. AUTHOR(s): (10) S. Kevin Counselman	(15) 8. CONTRACT OR GRANT NUMBER(s): DAAK40-79-D-0023
9. PERFORMING ORGANIZATION NAME AND ADDRESS: SCIENCE APPLICATIONS, INC. 2109 West Clinton Avenue, Suite 800 Huntsville, Alabama 35805	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: Task 011
11. CONTROLLING OFFICE NAME AND ADDRESS: Commander, U.S. Army Missile Command Attention: DRSMI-RPT Redstone Arsenal, Alabama 35898	(11) 12. REPORT DATE: September 1980
	13. NUMBER OF PAGES: 39
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Commander, U.S. Army Missile Command Attention: DRSMI-RGN Redstone Arsenal, Alabama 35898 (12) 43	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Text Editing
Computers
Sigma

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report examines the requirements and implementation of an interactive text editing facility for the Sigma 5.

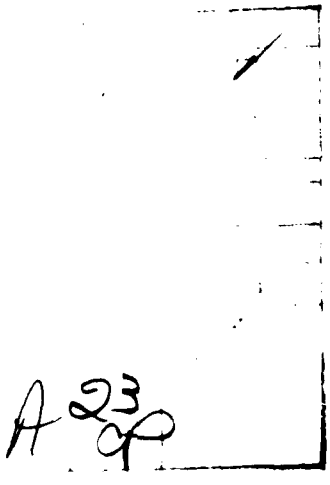
DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

19 X 21

TABLE OF CONTENTS



LIST OF TABLES

INTERACTIVE TEXT EDITING FOR THE SIGMA 5

I. INTRODUCTION

In the early 1960s, when systems such as the Sigma 5 were being released, most software development was done in a batch processing environment. Facilities for the interactive development of software were the exception rather than the rule.

Steadily declining hardware costs coupled with rising software development costs have created a new emphasis on improving the efficiency of software development utilities through interactive processing.

This report examines the requirements and implementation of one such utility, an interactive text editor for the Sigma 5.

II. EDITOR DESIGN AND IMPLEMENTATION

In general, a text editor reads symbolic information (be it source code, text or whatever) from some input file into an area of memory, performs some operation(s) on this information, and writes the result to an output file. A more detailed discussion of this process follows.

Input and Output Files. An editor input or output file is defined as a file, residing on magnetic disk, containing zero or more lines. A line is an 80-byte long record, corresponding to the standard 80 column punched card.

For the Sigma 5, under the RBM operating system, these requirements imply a compressed file format. The FSIZE parameter of the output file should be set to the expected number of lines in the workspace buffer at the conclusion of an edit session.

Consideration was given to using dynamic mass storage allocation to allow the size of the output file to vary during an edit session. However, since RBM requires a contiguous area for all disk files, there is no way to guarantee that sufficient disk space exists at the end of an arbitrary output file for dynamic expansion.

Workspace Buffer. In order to efficiently process the data read from the input file, this data must be placed in a memory resident workspace buffer. Since there is a relatively large access delay on a disk (due to seek and latency times), the workspace buffer should be as large as possible to maximize the amount of data transferred in a single disk access.

Coupled with this, however, is the constraint that the total program and buffer size cannot exceed available memory.

In a virtual memory system (e.g., DEC VAX 11/780), the available memory space is limited only by the amount of available secondary storage. The operating system pages information in and out of memory to maintain the portion of the buffer being operated on in memory.

In non-virtual systems, such as the Sigma 5, the available buffer space is limited to the amount of physical memory allocated to the user. If this memory space is too small (i.e., unable to contain the largest file which will be edited), a paging scheme similar to that of the virtual system must be implemented.

Typically, a set number of lines (page) is read into the workspace buffer. The user may then edit lines within the page. When a reference is made to a line beyond the end of the page, the current page is written to the output file and a new page is read from the input file. References to a line before the beginning of the page cause the remainder of the input file to be copied to the output file, and the output file is reopened as the new input file.

The system on which this editor was implemented had a 32K word block of memory available, which is sufficient for the program and a workspace buffer of approximately 1100 lines. Since the great majority of files at this facility were less than 1100 lines long, no paging facility was implemented.

The commands available to the user have a great influence on the design of the workspace buffer. The requirement that insertions and deletions be allowed at arbitrary points within the file, and that movement within the file (forward and backward) be unrestricted, dictates a doubly-linked list structure for the workspace buffer. A stack of available buffer locations should be maintained so that deleted lines may free their buffer space.

Edit operations

Any editing task may be specified in terms of one of three basic operations. These are:

(1) Insert a new line
(2) Delete an existing line
(3) Change an existing line

Additionally, commands must be provided to position to the desired line within the workspace buffer.

These commands form a functionally complete set of commands which can be used to perform any editing operation. Convenience of use can be greatly increased, however, with additional commands to do such things as FIND a particular string, MOVE particular lines from one place to another, and so on.

The commands incorporated into the current version of the editor for the Sigma 5 are shown in Table 1.

Currently, the specification of input and output files is performed by control commands sent to the RBM monitor. An effort is underway to allow the

TABLE 1. EDITOR COMMANDS

NOTE: Braces { } indicate an optional field. Reserved words are indicated by capital letters, lowercase letters indicate user-supplied fields.

CR indicates carriage return.

Ø indicates a required blank

I {NSERT} {Ø string} CR

If string is specified, it is inserted following the current line. Otherwise, an insert mode is entered, and all lines typed will be entered following the current line. The insert mode is exited by entering CR as the first character of a line.

D {ELETE} {n}

Deletes the curent and next n-1 lines. If n is not specified, the current line is deleted.

+ {n}

Advances the current position pointer n lines. If n is not specified, 1 is assumed.

- {n}

Backs up the current position pointer n lines. If n is not specified, 1 is assumed.

T {OP}

Moves current position pointer to top of workspace.

B {OTTOM}

Moves current position pointer to bottom of workspace.

P{ RINT } {n}

Prints current and next n-1 lines. If n is not specified, 1 is assumed.

C {HANGE} Ø delimeter old string delimeter new string delimeter

Changes first occurrence of old string in current line to new string. First non-blank character is taken to be the delimeter. If trailing delimeter is not specified, new string is asumed to extend to the end of the line.

TAB { tab_1, tab_2, ... tab_n }

Sets up to 11 tab positions. The command has no effect if no tabs are specified.

TABLE 2. SAMPLE JOB STREAM

```
! JOB  EDIT, IFILE
! STDLB (SI, D3, IFILE)                    (1)
! STDLB (SO, D3, OFILE)
! EDIT                                     (2)
! FIN
```

NOTES:

1. The STDLB command should be used to assign SI to the edit input file and SO to the edit output file.

2. In order for EDIT to be loaded into memory, the user must do an FMEM Ø Keyin.

files to be specified from the edit terminal (in this case a Tektronix 4002). The sample job stream for the current configuration is shown in Table 2.

Also under development are modifications which will allow the editor to run as a foreground program with interrupt driven I/O routines. This will allow background batch processing to run concurrently with edit I/O operations. Since most of the time used by an editor is in waiting for user response and in doing I/O, interrupt driven operation of the editor will have very little effect on the turnaround time for background jobs.

Appendix 1 shows a sample edit session using the current version of the editor.

Appendix 2 is a listing of the editor program.

III. CONCLUSIONS AND RECOMMENDATIONS

The interactive text editor described in this report has been installed on the Sigma 5 system located in the Guidance and Control Analysis facility at Redstone Arsenal.

Use of this processor has resulted in a significant decrease in time required to create and update files of symbolic data.

It is recommended that this processor be expanded to include capabilities such as to FIND a particular string, to MOVE a line or group of lines to some location, and a capability to define MACRO commands consisting of several edit commands.

These enhancements should reduce even further the amount of time required for symbolic data entry and maintenance.

REFERENCES

1. Xerox Sigma 5 Computer Reference Manual, Document No. 90 09 59E, October 1971.

2. Hybrid Remote Batch Monitor Operator/User Manual for Redstone Arsenal, Code Research Corporation, July 1971.

3. Xerox Real-Time Batch Monitor RBM Operations Manual, Document No. 90 15 81D, April 1971.

4. Xerox Real-Time Batch Monitor RBM Operations Reference Manual, Document No. 90 16 47D, October 1972.

5. Xerox Real-Time Batch Monitor RBM User's Guide, Document No. 90 16 53B, October 1972.

6. Hazel, Larry, Plotter, an Interactive Program Control and Plot Generation Package for the Sigma 5, Science Applications, Inc., Huntsville, AL, Report NO. SAI-78-763-HU, August 1977.

7. RSX-11 Utilities Procedures Manual, Digital Equipment Corporation, Maynard, MA. Order No. AA-5567B-TC, December 1977.

APPENDIX 1

SAMPLE EDIT SESSION

```
*BOF*
EDI>I
IN >THIS DEMONSTRATES THE INSERT MODE.
IN >WHEN AN INSERT COMMAND IS FOLLOWED
IN >ONLY BY A CARRIAGE RETURN THE INSERT
IN >MODE IS ENTERED. TO EXIT THE INSERT
IN >MODE ENTER A LINE CONSISTING ONLY
IN >OF A CARRIAGE RETURN.
IN >
EDI>TOP
*BOF*
EDI>PRINT 5
*BOF*
   >THIS DEMONSTRATES THE INSERT MODE.
   >WHEN AN INSERT COMMAND IS FOLLOWED
   >ONLY BY A CARRIAGE RETURN THE INSERT
   >MODE IS ENTERED. TO EXIT THE INSERT
EDI>
```

```
+2
    >OF A CARRIAGE RETURN.
EDI>CHANGE /CARRIAGE/HORSE AND BUGGY/
    >OF A HORSE AND BUGGY RETURN.
EDI>TOP
*BOF*
EDI>P 7
*BOF*
    >THIS DEMONSTRATES THE INSERT MODE.
    >WHEN AN INSERT COMMAND IS FOLLOWED
    >ONLY BY A CARRIAGE RETURN THE INSERT
    >MODE IS ENTERED. TO EXIT THE INSERT
    >MODE ENTER A LINE CONSISTING ONLY
    >OF A HORSE AND BUGGY RETURN.
EDI>-5
    >THIS DEMONSTRATES THE INSERT MODE.
EDI>DELETE
    >WHEN AN INSERT COMMAND IS FOLLOWED
EDI>TOP
```

```
*BOF*
EDI>P 3
*BOF*
    >WHEN AN INSERT COMMAND IS FOLLOWED
    >ONLY BY A CARRIAGE RETURN THE INSERT
EDI>BOTTOM
    >OF A HORSE AND BUGGY RETURN.
EDI>I THE INSERT COMMAND CAN ALSO BE
EDI>I USED IN A SINGLE LINE INSERTION MODE
EDI>I LIKE THIS.
EDI>-5
    >MODE IS ENTERED. TO EXIT THE INSERT
EDI>PRINT 5
    >MODE IS ENTERED. TO EXIT THE INSERT
    >MODE ENTER A LINE CONSISTING ONLY
    >OF A HORSE AND BUGGY RETURN.
    >THE INSERT COMMAND CAN ALSO BE
    >USED IN A SINGLE LINE INSERTION MODE
EDI>
```

```
(TO TERMINATE THE EDIT SESSION AND
WRITE THE WORKSPACE BUFFER TO THE OUTPUT
FILE USE THE END COMMAND)

EDI>END
*END*
```

APPENDIX 2

LISTING OF EDITOR PROGRAM

```
C*************************************************************************
C
C     EDIT --- INTERACTIVE TEXT EDITOR FOR THE SIGMA 5
C     AUTHOR --- JOHN SELMAN
C                JUNE 1980
C
C     THIS PROGRAM EDITS A SPECIFIED INPUT FILE TO A SPECIFIED OUTPUT
C     FILE ACCORDING TO COMMANDS INPUT FROM THE TEKTRONIX GRAPHICS CONSOLE.
C     THE PROGRAM RUNS IN FOREGROUND, WHICH ALLOWS EDITING TO BE PERFORMED
C     CONCURRENTLY WITH BACKGROUND PROCESSING.
C
C*************************************************************************
      IMPLICIT INTEGER (A-Z)
C
C     THE WORKSPACE BUFFER IS A DOUBLY LINKED LIST, VARIABLE HEAD POINTS TO THE
C     HEAD OF THE LIST, TAIL TO THE TAIL OF THE LIST, F_LINK(I) CONTAINS
C     THE FORWARD LINK FOR THE ITH NODE, AND B_LINK(I) THE BACKWARD
C     LINK FOR THE ITH NODE. NODE 1 OF THE LIST CONSISTS OF A CARD IMAGE, WHICH
C     IS STORED IN BUFFER(I,J),J=1,20.
C
C     ADDITIONALLY, A STACK OF FREE NODES (AVAIL) IS MAINTAINED, SO THAT
C     DELETIONS CAN FREE BUFFER LOCATIONS FOR LATER USE.
C
      COMMON /COM/  COMBUF(22),  CON,  POS, NCR, TABS(11)
      COMMON  BUFFER, FLINK, BLINK, HEAD, TAIL, AVAIL, EOF,
     1           LINE,  EOF,  NOW
      DIMENSION IFNAM(2), OFNAM(2)
      COMMON /IO/  IN, OUT, IAREA, IFNAM, OAREA, OFNAM
      DIMENSION BUFFER(20,1200),FLINK(1200),BLINK(1200),AVAIL(1200)
C     INITIALIZE LINE COUNTER
      LINE = 0
C     INITIALIZE CURRENT BUFFER POSITION
      NOW = 0
C     INITIALIZE TEKTRONIX
      CALL ST7611
      CALL READC
C     SET UP DEFAULT TAB POSITIONS DEFINED BY TABSTOP
      DO 20 I=8,80,8
   20 TABS(I/8) = I
      TABS(11) = 80
C     SET UP INITIAL TAB MODE
      CALL TABSTOP(TABS(1)+2,TABS(2)+2,TABS(3)+2,TABS(4)+2,
     1          TABS(5)+2,TABS(6)+2,TABS(7)+2,TABS(8)+2,
     2          TABS(9)+2,TABS(10)+2,TABS(11)+2)
C     SET UP INPUT AND OUTPUT FILES
      CALL INSET
C
C     READ PAGE INTO WORKSPACE BUFFER
C
      CALL BUFRD(BUFFER,F_LINK,B_LINK,EOF)
  100 CONTINUE
C
C     GET COMMAND FROM KEYB AND RETURN INDEX
C
      CALL GETCOM
C     PERFORM COMMAND
      GO TO (110,120,130,140,150,160,170,180,190,200,210,220),COM
C     ERROR, SHOULDN'T GET HERE
      PAUSE 1
C
  110 CONTINUE
      CALL INSERT
```

```
      GO TO 500
C
120   CONTINUE
      CALL DELETE
      GO TO 500
C
130   CONTINUE
      CALL PLUS
      GO TO 500
C
140   CONTINUE
      CALL MINUS
      GO TO 500
C
150   CONTINUE
      CALL TYPES
      GO TO 500
C
160   CONTINUE
      CALL BETTER
      GO TO 500
C
170   CONTINUE
      CALL PAGE
      GO TO 500
C
180   CONTINUE
      CALL END
      GO TO 500
C
190   CONTINUE
      CALL DEBUG
      GO TO 500
C
200   CONTINUE
      CALL PRINT
      GO TO 500
C
210   CONTINUE
      CALL CHANGE
      GO TO 500
220   CONTINUE
      CALL TAB
500   CONTINUE
      IF (COM .EQ. 8) STOP
C
C     GET NEXT COMMAND
      GO TO 100
      END
```

```
      SUBROUTINE INSET
C*****************************************************************************
C
C     SUBROUTINE INSET PROMPTS USER FOR INPUT AND OUTPUT FILES,
C     AND INITIATES VALIDITY CHECKS ON THESE FILES.  SUBROUTINE WILL
C     LOOP UNTIL A CORRECT INPUT AND OUTPUT FILE HAS BEEN ENTERED.
C
C     CALLED BY:  MAIN
C     CALLS:      OUTSCOPE,  CHECKF
C
C*****************************************************************************
      COMMON /IO/  IN, OUT, IAREA, IFNAM, OAREA, OFNAM
      IMPLICIT INTEGER (A-Z)
C     IFNAM WILL CONTAIN THE INPUT FILE NAME, OFNAM THE OUTPUT FILE NAME
      DIMENSION IFNAM(2), OFNAM(2)
C     VARIABLES IAREA AND OAREA WILL CONTAIN THE INFILE AREA
C     AND THE OUTFILE AREA, RESPECTIVLY.
C
   10 CONTINUE
C     PROMPT FOR INPUT FILE
      CALL PROMPT ('ENTER INPUT FILE >')
C     READ REPLY AND VERIFY CORRECT AREA, FORMAT AND ACCESS
      CALL CHECKF(IAREA,IFNAM)
      IF (IAREA .EQ. AHERR) GO TO 10
   20 CONTINUE
C     PROMPT FOR OUTPUT FILE
      CALL PROMPT ('ENTER OUTPUT FILE >')
C     READ REPLY AND VERIFY CORRECT AREA, FORMAT AND ACCESS
      CALL CHECKF (OAREA,OFNAM)
      IF (OAREA .EQ. AHERR) GO TO 20
      RETURN
      END
```

```
      SUBROUTINE BUFRD
C*****************************************************************
C     SUBROUTINE BUFRD READS A PAGE FROM THE DISK FILE ASSOCIATED WITH UNIT
C     IN INTO THE WORKSPACE BUFFER, ADDITIONALLY THE LINKS FOR THE BUFFER
C     ARE SET UP, AND THE FREE NODES ARE PUT ON THE AVAIL STACK.
C
C
C     CALLED BY:  MAIN
C     CALLS:      BUFFERIN
C
C*****************************************************************
      IMPLICIT INTEGER (A-Z)
      COMMON /IO/  IN, OUT, IAREA, IFNAM, OAREA, OFNAM
      COMMON  BUFFER, FLINK, BLINK, HEAD, TAIL, AVAIL, TOP,
     1           LINE,  EOF,  NOW
      DIMENSION BUFFER(20,1200),FLINK(1200),BLINK(1200),AVAIL(1200)
C     GET MAX LINES IN BUFFER
C     FOR PRODUCTION VERSION, WILL BE END OF MEMORY (10000 HEX) - FIEND
C     DIVIDED BY 20 WORDS PER LINE (20 DATA, 2 LINK, 1 STACK), MINUS EXPANSION
C     AREA.  FOR NOW, ASSUME BUFFER , LINKS AND STACK ARE AS DECLARED IN
C     DIMENSION STATEMENT, AND THAT THE PAGE SIZE IS 1100 LINES
C
C
C***  TEMP DEBUG CODE  *********
      IN = 2
C***  END DEBUG  *********
      EOF = 0
C     READ IN PAGE FROM UNIT IN
C
      DO 7 N=1,1100
         CALL BUFFERIN(IN,1,BUFFER(1,N),20,STAT)
         IF (STAT .EQ. 3)  EOF = 1;  GO TO 9
7        CONTINUE
9     CONTINUE
      IF (EOF .EQ. 1) N = N - 1
      TOP = 0
C     PUT UNUSED NODES ON THE STACK
      DO 10 I=N+1, 1100
      TOP = TOP + 1
      AVAIL(TOP) = I
10    CONTINUE
C     IF N IS ZERO  BUFFER IS EMPTY
      REPEAT 15 WHILE N .EQ. 0
         HEAD = 0
         TAIL = 0
         RETURN
15       CONTINUE
C
C     SET UP LINKS
      DO 20 I=1,N-1
      FLINK(I) = I+1
      BLINK(I) = I-1
20    CONTINUE
      FLINK(N) = -1
      BLINK(N) = N-1
      HEAD = 1
      TAIL = N
      END
```

```
      SUBROUTINE GETCOM
C*****************************************************************************
C     SUBROUTINE READS COMMAND FROM TEKTRONIX, AND ATTEMPTS TO MATCH FIRST
C     TOKEN WITH ENTRYS IN THE COMMAND TABLE.
C     CALLED BY:  MAIN
C     CALLS:      PROMPT,  INSCOPE,  GETCHR
C*****************************************************************************
      IMPLICIT INTEGER (A-Z)
      COMMON /CMD/  CMBUF(22),  COM,  POS, NCR, TABS(11)
      DIMENSION TABLE(10,2)
      DATA TABLE          /'I*INSERT*',
     1                     'D*ELETE*',
     2                     '***     ',
     3                     '---     ',
     4                     'T*OP*   ',
     5                     'B*OTTOM*',
     6                     'PAGE**  ',
     7                     'END**   ',
     8                     'DEBUG** ',
     9                     'P*RINT* ',
     A                     'C*HANGE*',
     B                     'TAB**   '/
      DATA NUMCOMS  /12/
C*****************************************************************************
      CALL PROMPT ('EDI>',4)
10    CONTINUE
      CALL INSCOPE (CMBUF,20,NCR)
      CMBUF(21)=CMBUF(22)= 44
      IF (NCR .NE. 0) GO TO 12
      POS=2
      CMBUF(1) = 44
12    CONTINUE
C
C     INNER LOOP SCANS INPUT CHARS AGAINST CHARS FROM VALID COMMAND
C     TABLE.  OUTER LOOP CONTROLS WHICH ENTRY COMPARING AGAINST.  VARIABLE
C     MODE IS INCREMENTED WHENEVER AN ASTERISK IS ENCOUNTERED IN THE
C     TABLE, AND IS INTERPRETED AS FOLLOWS:
C                 MODE=0   CHARS THUS FAR ARE NOT SUFFICIENT TO DEFINE THIS CMD
C                 MODE=1   CHARS THUS FAR ARE SUFFICIENT BUT NOT COMPLETE
C                 MODE=2   ALL CHARS IN THE COMMAND HAVE BEEN SCANNED
C
        DO 30 COM=1, NUMCOMS
        MODE = 0
        DO 20 POS=1, 8
C
C     GET CHAR FROM TABLE, IF * INC MODE AND GET NEXT
15      CONTINUE
        X = GETCHR(TABLE,COM,POS+MODE)
        IF (X .EQ. 1H*) MODE = MODE+1   GO TO 15
C
C     GET CHAR FROM COMMAND BUFFER
C
        Y = GETCHR(CMBUF,0,POS)
C
C     IF Y IS A BLANK OR A NUMBER, AND MODE > 0, WE ARE DONE
C
        IF((MODE.GT.0).AND.((Y.EQ.1H ).OR.((Y.GE.1H0).AND.(Y.LE.1H9))))
     1                  RETURN
C
C     IF MODE=2 THIS COMMAND IS TOO LONG, CHECK NEXT CMD
        IF (MODE .EQ. 2) GO TO 30
C
```

```
C     IF THIS CHAR DOESNT MATCH TABLE ENTRY, TRY NEXT CMD
        IF (X .NE. Y)  GO TO 30
C
C     LOOP FOR NEXT CHARACTER (NEXT POS)
C
   20   CONTINUE
C
C     LOOP FOR NEXT COMMAND  (NEXT CMD)
C
   30   CONTINUE
C
C     IF WE FALL THRU TO HERE, WE HAD AN ILLEGAL CMD. PROMPT AND TRY AGAIN
C
        CALL OUTSCOPE ('ILLEGAL COMMAND', REENTER,0,25)
        GO TO 10
        END
```

```
1.            SUBROUTINE TAB
2.            IMPLICIT INTEGER (A-Z)
3.            COMMON /COM/ COMBUF(32), COM, POS, VAR, TABS(11)
4.            DECODE (74,10,COMBUF) POS,(TABS(I),I=1,11)
5.      10    FORMAT (VX,11I)
6.            CALL TABST2(TABS(1)+2,TABS(2)+2,TABS(3)+2,TABS(4)+2,
7.           1            TABS(5)+2,TABS(6)+2,TABS(7)+2,TABS(8)+2,
8.           2            TABS(9)+2,TABS(10)+2,TABS(11)+2)
9.            END
```

```
      SUBROUTINE INSERT
C*****************************************************************************
C
C     SUBROUTINE INSERT INSERTS THE LINE PASSED IN THE COMMAND BUFFER (CBUF)
C     INTO THE DOUBLY LINKED LINE BUFFER (BUFFER).  THE INSERTION OCCURS AFTER
C     THE CURRENT LINE POSITION (POINTED TO BY NOW).  THE SUBROUTINE SETS LINKS
C     THE HEAD AND/OR THE TAIL TO REFLECT THE INSERTION.
C
C*****************************************************************************
      IMPLICIT INTEGER (A-Z)
      COMMON /CMD/  CBUF(22),  CNT,  POS, NCR, TABS(11)
      COMMON  BUFFER, FLINK, BLINK, HEAD, TAIL, AVAIL, TOP,
     1           LINE,  EOF,  NOW
      DIMENSION BUFFER(20,1200),FLINK(1200),BLINK(1200),AVAIL(1200)
C
C     IF NCR=POS = 1 THEN USER DID A '<CR>', THIS PUTS
C     US INTO THE INPUT MODE. INPUT MODE IS EXITED BY '<CR>',I.E.
C     AN EMPTY LINE (NCR=0).
C
      IMODE = 0
      IF (POS-NCR .NE. 1)  GO TO 5
      IMODE = 1
      CALL PROMPT ('IN >',4)
      CALL TABSTOP(TABS(1),TABS(2),TABS(3),TABS(4),TABS(5),TABS(6),
     1           TABS(7),TABS(8),TABS(9),TABS(10),TABS(11))
5     CONTINUE
      IF (IMODE .EQ. 1) CALL INSCOPE(CBUF,CD,NCR)
     1                        CBUF(21)=CBUF(22)=4H    ;POS=0
      REPEAT GO, WHILE NCR .GT. 0
C
C     INSERT LINE INTO NEXT AVAILABLE BUFFER SLOT
C
      IF (TOP .LE. 0) CALL OUTSCOPE('WORKSPACE FULL',14)  RETURN
      I = AVAIL(TOP)
      TOP = TOP - 1
C
C     IN-LINE CODE TO MOVE BYTES FROM CBUF TO BUFFER
S     LI,2      79              COMPUTE BYTE INDEX
S     AW,2      POS             INTO CBUF
S     LW,3      I               COMPUTE BYTE INDEX
S     AI,3      -1              INTO BUFFER TO GET
S     MI,3      80              TO THE 80TH ENTRY
S     AI,3      79              COPY FROM RIGHT TO LEFT
S10   LB,1      CBUF,2          GET BYTE FROM CBUF
S     STB,1     BUFFER,3        STASH IN BUFFER
S     AI,2      -1              DECREMENT CBUF INDEX
S     AI,3      -1              DECREMENT BUFFER INDEX
S     CW,2      POS             DONE
S     BGE       10S             NOT YET
C     INSERTION COMPLETE
C     SET UP LINKS
C
C     CASE 1:  IF HEAD = 0, THE LIST IS EMPTY
      IF (HEAD .NE. 0) GO TO 20
      HEAD = I
      TAIL = I
      BLINK(I) = 0
      FLINK(I) = 0
      NOW = I
      LINE = I
      GO TO 45
20    CONTINUE
```

```
C
C     CASE 2:  IF N9W = 0  WE ARE BEFORE THE HEAD
C
        IF (N9W .NE. 0)  GO TO 30
        FLINK(I) = HEAD
        BLINK(I) = 0
        BLINK(HEAD) = I
        HEAD = I
        N9W = I
        LINE = 1
        GO TO 65
30      CONTINUE
C
C     CASE 3:  WE ARE SOMEWHERE WITHIN THE LIST, POSSIBLY AT THE TAIL
C
        BLINK(I) = N9W
        FLINK(I) = FLINK(N9W)
        IF (TAIL .NE. N9W)  BLINK (FLINK(N9W) ) = I
        FLINK(N9W) = I
40      CONTINUE
        IF (TAIL .EQ. N9W)  TAIL = I
        N9W = I
        LINE = LINE + 1
65      IF (LINDE .EQ. 0)  RETURN
        CALL INSCORE(C9YBUF,20,NCR)
        C9YBUF(21)=C9YBUF(22)+44
        P9S = 0
50      CONTINUE
        CALL PRMPT ('ED[>'+6)
        CALL TABSTOP(TABS(1)+2,TABS(2)+2,TABS(3)+2,TABS(4)+2,
     1                 TABS(5)+2,TABS(6)+2,TABS(7)+2,TABS(8)+2,
     2              TABS(9)+2,TABS(10)+2,TABS(11)+2)
        END
```

```
      SUBROUTINE DELETE
C*************************************************************************
C
C     SUBROUTINE DELETES N NODES FROM DOUBLY-LINKED BUFFER AND
C     RETURNS NODES TO AVAIL STACK.
C
C*************************************************************************
      IMPLICIT INTEGER (A-Z)
      COMMON /COM/  COMBUF(22),  COM,  POS, NCR, TABS(11)
      COMMON  BUFFER, FLINK, BLINK, HEAD, TAIL, AVAIL, TOP,
     1           LINE,  EOF,  NOW
      DIMENSION BUFFER(20,1200),FLINK(1200),BLINK(1200),AVAIL(1200)
C
C     GET NUMBER OF LINES TO DELETE INTO NUM
C
      DECODE (12,10,COMBUF) POS+1, NUM
10    FORMAT (NX,I)
C
C     ERR DELETES NUM LINES
C
      DO 200 I=1,NUM
C
C     IF LIST IS EMPTY (HEAD.EQ.TAIL.EQ.NOW.0) OR WE ARE AT 'EOF' (NOW=0)
C     PRINT 'EOF' AND RETURN
C
      IF (NOW .EQ. 0)  CALL OUTCOPE('*EOF*',5)  RETURN
C
C     PLACE THE NODE ABOUT TO BE DELETED ON THE AVAIL STACK
C
      TOP = TOP + 1
      AVAIL(TOP) = NOW
C
C     IF LIST HAS ONLY ONE NODE (HEAD.EQ.TAIL.EQ.NOW), OR IF WE ARE DELETING
C     THE TAIL NODE (TAIL=NOW), WE WILL PRINT 'EOF'. IF THERE IS ONLY
C     ONE NODE, HEAD, TAIL, AND NOW WILL BE SET TO 0.AND RETURN.
C     OTHERWISE TAIL AND NOW WILL BE SET TO BLINK(NOW) AND LINE NOW WILL
C     BE PRINTED.
C
      REPEAT 100, WHILE NOW .EQ. TAIL
      CALL OUTCOPE ('*EOF*',5)
      OLD = NOW
      TAIL = NOW = BLINK(NOW)
      IF (HEAD .EQ. OLD)  HEAD = 0  RETURN
      FLINK(NOW) = -1
      CALL PRINT( BUFFER(1,NOW))
      RETURN
100   CONTINUE
C
C     WE ARE ABOUT TO DO A DELETE
C     WE ARE ABOUT TO DO A DELETION WHICH IS NOT AT THE TAIL,
C     BUT MAY BE AT THE HEAD).  WE WILL SET UP LINKS, SET NOW=FLINK,
C
      FLINK( BLINK(NOW)) = BLINK(NOW)
      IF (BLINK(NOW) .EQ. 0)  HEAD = FLINK(NOW)
      IF (BLINK(NOW) .NE. 0)  FLINK(BLINK(NOW)) = FLINK(NOW)
      NOW = FLINK(NOW)
C
C     DO NEXT DELETION
C
200   CONTINUE
      CALL PRINT (BUFFER(1,NOW))
      END
```

```
      SUBROUTINE PLUS
      IMPLICIT INTEGER (A-Z)
      COMMON /COM/  COMBUF(22),  COM,  BOS, NCR, TABS(11)
      COMMON  BUFFER, FLINK, BLINK, HEAD, TAIL, AVAIL, TOP,
     1            LINE, EOF, NOW
      DIMENSION BUFFER(20,1200),FLINK(1200),BLINK(1200),AVAIL(1200)
      DECODE (6,10,COMBUF) BOS-1, NUM
   10 FORMAT (Hx,I)
C     IF LIST IS EMPTY, PRINT *EOF* AND RETURN
      IF (HEAD .EQ. 0) CALL OUTSCOPE('*EOF*',5)  RETURN
      DO 20 I=1,NUM
      IF (NOW .EQ. 0)  NOW = HEAD;  GO TO 20
      IF (NOW .EQ. TAIL) GO TO 30
      NOW = FLINK(NOW)
   20 CONTINUE
C     DESIRED LINE LOCATED, NOW PRINT IT
   25 CONTINUE
      CALL PRINT_BUFFER(1,NOW))
      RETURN
   30 CONTINUE
C     END OF LIST ENCOUNTERED
      CALL OUTSCOPE('*EOF*',5)
      GO TO 25
      END
```

```
      SUBROUTINE MINUS
      IMPLICIT INTEGER (A-Z)
      COMMON /COMD/  COMBUF(22),  C94,  PBS, NCR, TABS(11)
      COMMON  BUFFER, FLINK, BLINK, HEAD, TAIL, AVAIL, TOP,
     1           LINE, FBF, NDW
      DIMENSION BUFFER(20,1200),FLINK(1200),BLINK(1200),AVAIL(1200)
      DECODE (6,10,COMBUF) PBS-1,  NDW
   10 FORMAT (Nx,I)
      IF (NDW .EQ. 0)  30 TO 30
      DO 20 I=1,NDW
      NDW = BLINK(NDW)
      IF (NDW .EQ. 0) GO TO 30
   20 CONTINUE
C     DESIRED LINE LOCATED, NOW PRINT IT
      CALL PRINT_(BUFFER(1,NDW))
      RETURN
   30 CONTINUE
C     BEGINNING OF LIST ENCOUNTERED
      CALL OUTSTR (1HBOF,1,5)
      END
```

```
      SUBROUTINE TEST3
      IMPLICIT INTEGER (A-Z)
      COMMON  BUFFER, FLINK, BLINK, HEAD, TAIL, AVAIL, TSP,
     1          LINE,  EOF,  N94
      DIMENSION BUFFER(20,1200),FLINK(1200),BLINK(1200),AVAIL(1200)
      N94 = 0
      CALL BUTSCOPE(1,BUFFER,5)
      END
```

```
      SUBROUTINE BOTTOM
      IMPLICIT INTEGER (A-Z)
      COMMON /BUFFER/ FLINK, BLINK, HEAD, TAIL, AVAIL, EOF,
     1                LINE, ERF, N94
      DIMENSION BUFFER(20,1200),FLINK(1200),BLINK(1200),AVAIL(1200)
      N94 = TAIL
C     IF LIST WAS EMPTY PRINT *BOF* AND RETURN
      IF (N94 .EQ. 0) CALL OUTSCOPE('*BOF*',5)  RETURN
      CALL PRINT_(BUFFER(1,N94))
      END
```

```
1:          SUBROUTINE PAGE
2:          CALL SUTSCOPE (1,PAGE=0,0,6)
3:          END
```

```
      SUBROUTINE END
      IMPLICIT INTEGER (A-Z)
      COMMON BUFFER, FLINK, BLINK, HEAD, TAIL, AVAIL, TRP,
     1           LINE, EOF, NOW
      DIMENSION BUFFER (20,1200),FLINK(1200),BLINK(1200),AVAIL(1200)
      COMMON /IO/  IN, OUT, IAREA, IFNAM, OAREA, OFNAM
      DIMENSION IFNAM(3), OFNAM(2)
C
C     COPY BUFFER TO OUTFILE BY TRAVERSING THE LIST.
C
C
C     TEMP CODE
      OUT = 3
      REWIND OUT
C     END TEMP CODE ******
      IF (HEAD .LE. 0)  RETURN
      NOW = HEAD
      REPEAT 20, WHILE NOW .NE. -1
        CALL BUFFEROUT(OUT,1,BUFFER(1,NOW),20,S)
        IF(S.EQ.3) CALL OUTSCOPE('*** EOF ON OUTFILE ***',22)/RETURN
        IF(S.EQ.4)CALL OUTSCOPE('***ERROR ON WRITE TO OUTFILE***',31)
        NOW = FLINK(NOW)
20    CONTINUE
      ENDFILE  OUT
      CALL OUTSCOPE ('END',5)
      END
```

```
      SUBROUTINE DEBUG
      IMPLICIT INTEGER (A-Z)
      COMMON /CMD/  C99BUF(22),  T94,  P9S, NCR, TABS(11)
      COMMON  BUFFER, FLINK, BLINK, HEAD, TAIL, AVAIL, T9P,
     1            LINE,  E9F,  N94
      COMMON /IN/  IN, SUT, IAREA, IFNAM, OAREA, OFNAM
      DIMENSION BUFFER(20,1200),FLINK(1200),BLINK(1200),AVAIL(1200)
      DIMENSION IFNAM(2), OFNAM(2)
      DIMENSION NUM(2)
      SUB = N94
    5 CONTINUE
      SET = 0
      CALL PRMPT ('PRINT, SET, OR EXIT  >',2)
      CALL INSCOPE(C99BUF,2)
      IF (C99BUF(1) .EQ. 64EXIT)  RETURN
      IF (C99BUF(1) .EQ. 64SET ) SET = 1
      CALL PRMPT ('ENTER VARIABLE >',15)
      CALL INSCOPE (C99BUF,2)
      IF (SET .EQ. 1) GO TO 10
      IF (C99BUF(1).EQ.64FLINK) CALL D2(FLINK(SUB));  GO TO 5
      IF (C99BUF(1).EQ.64BLINK) CALL D2(BLINK(SUB));  GO TO 5
      IF (C99BUF(1).EQ.64HEAD) CALL D2(HEAD);  GO TO 5
      IF (C99BUF(1).EQ.64TAIL ) CALL D2(TAIL );  GO TO 5
      IF (C99BUF(1).EQ.64T9P ) CALL D2(T9P );  GO TO 5
      IF (C99BUF(1).EQ.64LINE) CALL D2(LINE);  GO TO 5
      IF (C99BUF(1).EQ.64E9F ) CALL D2(E9F );  GO TO 5
      IF (C99BUF(1).EQ.64N94 ) CALL D2(N94 );  GO TO 5
      IF (C99BUF(1).EQ.64SUB ) CALL D2(SUB );  GO TO 5
      IF (C99BUF(1).EQ.64TABS) CALL D2(TABS(SUB));GO TO 5
      CALL SUTSCOPE ('ILLEGAL VARIABLE',16);  GO TO 5
    7 SET CODE
   10 CONTINUE
      CALL PRMPT ('ENTER SET VALUE >',17)
      CALL INSCOPE (NUM,2)
      IF(C99BUF(1).EQ.64FLINK) DECODE(8,77,NUM   ) FLINK(SUB);  GO TO 5
      IF(C99BUF(1).EQ.64BLINK) DECODE(8,77,NUM   ) BLINK(SUB);  GO TO 5
      IF(C99BUF(1).EQ.64HEAD) DECODE(8,77,NUM   ) HEAD;  GO TO 5
      IF(C99BUF(1).EQ.64TAIL) DECODE(8,77,NUM   ) TAIL;  GO TO 5
      IF(C99BUF(1).EQ.64T9P ) DECODE(8,77,NUM   ) T9P ;  GO TO 5
      IF(C99BUF(1).EQ.64LINE) DECODE(8,77,NUM   ) LINE;  GO TO 5
      IF(C99BUF(1).EQ.64E9F ) DECODE(8,77,NUM   ) E9F ;  GO TO 5
      IF(C99BUF(1).EQ.64N94 ) DECODE(8,77,NUM   ) N94 ;  GO TO 5
      IF(C99BUF(1).EQ.64SUB ) DECODE(8,77,NUM   ) SUB ;  GO TO 5
      IF(C99BUF(1).EQ.64TABS) DECODE(8,77,NUM   ) TABS ;  GO TO 5
      CALL SUTSCOPE('ILLEGAL VARIABLE',16)
      GO TO 5
   77 FORMAT (I)
      ENTRY SUBROUTINE D2(IVAR)
      ENCODE (8,77,NUM) IVAR
      CALL SUTSCOPE (NUM,8)
   77 FORMAT (' ',I)
      END
```

```
      SUBROUTINE PRINT
C*****************************************************************
C     SUBROUTINE PRINT OUTSCROLLS THE CURRENT LINE AND THE NEXT N-1 LINES.
C     IF THE CURRENT POSITION IS AT EOF, THE TRANSITION FROM EOF
C     TO LINE 1 COUNTS AS 1 LINE.  IF THE BUFFER IS EMPTY, PRINT OUTPUTS
C     A EOF, AND RETURNS
C*****************************************************************
      IMPLICIT INTEGER (A-Z)
      COMMON /CHAR/ COMBUF(22),  CON,  POS, NCR, TABS(11)
      COMMON  BUFFER, FLINK, BLINK, HEAD, TAIL, AVAIL, TOP,
     1        LINE, EOF, N94
      DIMENSION BUFFER(20,1200),FLINK(1200),BLINK(1200),AVAIL(1200)
C
C
C     IF LIST IS EMPTY, PRINT EOF AND RETURN
      IF (HEAD .EQ. 0) CALL OUTSCOPE('EOF',5)  RETURN
C
      DECODE (12,10,COMBUF) POS-1, N94
10    FORMAT (N4,I)
      DO 20 I = 1, N94
      IF (( N94 .EQ. 0) .AND. (I .GT. 1)) N94=HEAD; GO TO 18
      REPEAT 15 WHILE N94 .EQ. 0
         CALL OUTSCOPE('EOF',5)
         GO TO 20
15       CONTINUE
      IF (I .NE. 1)  N94 = FLINK(N94)
      REPEAT 17 WHILE N94 .EQ. -1
         CALL OUTSCOPE('EOF',5)
         N94 = TAIL
         CALL PRINTL(BUFFER(1,N94))
         RETURN
17       CONTINUE
18    CALL PRINTL(BUFFER(1,N94))
20    CONTINUE
      END
```

```
      SUBROUTINE PRINTL (BUF)
      IMPLICIT INTEGER (A-Z)
      DIMENSION PBUF(21), BUF(20)
C
C     ROUTINE OUTPUTS BLANK BLANK BLANK > BUF   TO SCOPE
C
      PBUF(1) = 44   >
      J = 84
      DO 10 I=20,1,-1
      IF (J .GT. 0) GO TO 5
      IF (BUF(I) .EQ. 44     ) J=J+4) GO TO 10
      J = -J
5     CONTINUE
      PBUF(I+1) = BUF(I)
10    CONTINUE
      CALL OUTSCOPE(PBUF,J)
      END
```

```
SUBROUTINE CHECKF
END
```

```
      SUBROUTINE CHANGE
C*****************************************************************
C
C     SUBROUTINE CHANGES OLDSTRING TO NEWSTRING. SYNTAX IS AS FOLLOWS:
C     C(HANGE) /OLDSTRING/NEWSTRING/
C     ADDITIONALLY, ANY CHARACTER CAN BE USED AS A DELIMETER FOR THE STRINGS
C     ( SHOWN AS / IN SYNTAX).
C
C     IMPLEMENTATION NOTES FOLLOW.
C     OCUR1     MARKS THE BEGINNING OF OLDSTRING IN COMMAND BUFFER
C     OCUR2     SCANS OLDSTRING IN COMMAND BUFFER
C     BCUR1     MARKS BEGINNING OF CANDIDATE MATCH STRING IN CURRENT LINE.
C     BCUR2     SCANS CURRENT LINE FOR MATCH CHECK
C
C     AFTER MATCH THESE VARIABLES ARE USED FOR DATA MOVEMENT, ETC..
C
C*****************************************************************
      IMPLICIT INTEGER (A-Z)
      COMMON /CMD/  CBUF(22),  CP,  PBS, NCR, TABS(11)
      COMMON  BUFFER, FLINK, BLINK, HEAD, TAIL, AVAIL, TOP,
     1          LINE, EOF, NEW
      DIMENSION TEMP(20)
      DIMENSION BUFFER(20,1200),FLINK(1200),BLINK(1200),AVAIL(1200)
C
C     FIRST NON-BLANK WILL BE DELIMETER
C
      DELIM = GETCHR(CBUF,0,PBS)
      REPEAT 5, WHILE DELIM .EQ. 1H
        PBS = PBS + 1
      DELIM = GETCHR(CBUF,0,PBS)
5     CONTINUE
      OCUR1 = PBS + 1
      BCUR1 = 0
      MATCH = 0
      REPEAT 30, WHILE MATCH .EQ. 0
        BCUR1 = BCUR1 + 1
        C2 = GETCHR(CBUF,0,OCUR1)
        REPEAT 10, WHILE GETCHR(BUFFER(1,NEW),0,BCUR1) .NE. C2
          BCUR1 = BCUR1 + 1
          IF (BCUR1 .GE. 81) CALL OUTSCOPE(INS FIND(,7))  RETURN
10      CONTINUE
C       WE NOW HAVE A MATCH ON FIRST CHARS, CHECK FOR MATCH ON THE
C       REST  IF NO MATCH THEN REPEAT 30 AGAIN
        B2 = 0
        C2 = 0
        BCUR2 = BCUR1 + 1
        OCUR2 = OCUR1 + 1
        REPEAT 20, WHILE B2 .EQ. C2
          C2 = GETCHR(CBUF,0,OCUR2)
          B2 = GETCHR(BUFFER(1,NEW),0,BCUR2)
          BCUR2 = BCUR2 + 1
          OCUR2 = OCUR2 + 1
20      CONTINUE
C       IF C2 IS DELIM THEN WE HAVE A MATCH
        IF (C2 .EQ. DELIM)  MATCH = 1
30    CONTINUE
C
C     WE HAVE A MATCH, FROM BCUR1 TO BCUR2-2
C     COPY CHARS 1 THRU BCUR1-1 TO TEMP BUFFER
C
      REPEAT 40, FOR I=(1,BCUR1-1)
        T = GETCHR(BUFFER(1,NEW),0,I)
```

```
C
C       INLINE CODE TO PUT T IN INTO BYTE OF TEMP
C
        L4,1    I
        AI,1    -1          ZERO INDEXING
        L3,2    T
        STB,2   TEMP,1      STASH IN TEMP
C
50      CONTINUE
C
C       COPY CHARS FROM NEWSTRING INTO TEMP
        I = BCUR1 - 1
        T = GETCHR(CBMBUF,0,CCUR2)
        REPEAT 60, WHILE T .NE. DELIM
C
C       INLINE CODE TO STORE T INTO THE INTO BYTE OF TEMP
C
C       L4,1    I
C       L3,2    T
C       STB,2   TEMP,1
C
          I = I + 1
          IF (I .GE. 80)  GO TO 75
          CCUR2 = CCUR2 + 1
          T = GETCHR(CBMBUF,0,CCUR2)
60      CONTINUE
C
C       COPY CHARS FROM BCUR2-1 OF BUFFER TO TEMP
C
        J = BCUR2 - 1
        REPEAT 60, WHILE (.LT. 80 .AND. J .LE. 80
          T = GETCHR(BUFFER(1,NOW),0,J)
C
C       INLINE CODE TO STORE T INTO BYTE I
C
C       L4,1    I
C       L3,2    T
C       STB,2   TEMP,1
          I = I + 1
          J = J + 1
60      CONTINUE
C
C       IF I IS STILL < 80 WE NEED TO BLANK FILL ON RIGHT
C
        REPEAT 70 WHILE I .LT. 80
C
C       INLINE CODE TO STORE BLANK IN BYTE I
C
C       L4,1    I
C       LI,2    X'40'
C       STB,2   TEMP,1
C
          I = I + 1
70      CONTINUE
75      CONTINUE
C
C       NOW MOVE TEMP BUFFER BACK INTO CURRENT LINE
C
        DO 80 I=1,80
          BUFFER(I,NOW) = TEMP(I)
80      CONTINUE
        CALL PRINT(BUFFER(1,NOW))
C       THATS ALL FOLKS
```

DISTRIBUTION

	No. of Copies
Commander Defense Documentation Center ATTN: DDC-TCA Cameron Station Alexandria, VA 22314	12
Commander US Army Research Office ATTN: DRXRO-PH, Dr. R. Lontz P.O. Box 12211 Research Triangle Park, NC 27709	5
US Army Research & Standardization Group (Europe) ATTN: DRXSN-E-RX, Dr. Alfred K. Kedoluha Box 65 FPO New York 90510	1
Commander US Army Material Development & Readiness Command ATTN: Dr. James Bender Dr. Gordon Bushy 5001 Eisenhower Avenue Alexandria, VA 22333	1
Hq, Department of the Army Office of the DCS for Research, Development & Acquisition ATTN: DAMA-ARZ Room 3A474, The Pentagon Washington, DC 20310	1
OUSDR&E ATTN: Mr. Leonard R. Weisberg Room 301079, The Pentagon Washington, DC 20301	1
Director Defense Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, VA 22209	1
OUSDR&E ATTN: Dr. G. Ganota Deputy Asst. for Research (Research in Advanced Technology) Room 301067, The Pentagon Washington, DC 20301	1

DISTRIBUTION (Concluded)

		No. of Copies
DRSMI-R	Col. Fiest	1
-RG	Mr. Todd	1
	Mr. Plunkett	1
-RGN	Mr. Murdock	5
	Dr. Pastrick	1
	Dr. Kelly	1
-RPT		2
-RR	Dr. Hartman	1
	Dr. Bennett	1
DRSMI-LP	Mr. Voigt	1
IIT Research Institute ATTN: GACIAC 10 West 35th Street Chicago, Illinois 60616		1